A Model-based Approach to the Automatic Revision of Secondary Legislation

Tingting Li\textsuperscript{1}  Tina Balke\textsuperscript{2,1}  Marina De Vos\textsuperscript{1}  Julian Padget\textsuperscript{1}  Ken Satoh\textsuperscript{3}

\textsuperscript{1}University of Bath  \\ \{t.li,mdv,jap\}@cs.bath.ac.uk

\textsuperscript{2}University of Surrey  \\ t.balke@surrey.ac.uk

\textsuperscript{3}National Institute of Informatics  \\ ksatoh@nii.ac.jp

The 14th International Conference on AI and Law (ICAIL 2013)
1 Motivation

2 Modelling of Legal Specification

3 Automatic Conflict Detection

4 Conflict Resolution via Automatic Revision

5 Future Work
Conflict of Laws

- Law is dynamic: adding new laws or amending existing ones.
- Existing inferior law (or set of regulations) may no longer be consistent with the new superior law: *an action permitted (obliged) in one but prohibited in another.*
- Possibly resulting in (unintended) illegal behaviour.
- **Problem:** detection and resolution of such conflicts automatically.
- In particular: Will a specific scenario (a course of actions) result in a conflict? How to revise existing laws to remove this conflict?
Conflict of Laws

- Law is dynamic: adding new laws or amending existing ones.
- Existing inferior law (or set of regulations) may no longer be consistent with the new superior law: *an action permitted*(obliged) in one but *prohibited in another*.
- possibly resulting in (unintended) illegal behaviour.
- **Problem**: detection and resolution of such conflicts automatically.
- In particular: Will a specific scenario (a course of actions) result in a conflict? How to revise existing laws to remove this conflict?
Case Study

- Recent change in UK immigration law about student visa regulation\(^1\).
- **UK Immigration Law**: the permitted working hours of overseas students are reduced: *up to 20 hours per week*.
- **Studentship Regulations**: minimum number of teaching hours the student has to deliver: *at least 30 hours per week*.

---

Recent change in UK immigration law about student visa regulation \(^1\).

**UK Immigration Law** : the permitted working hours of overseas students are reduced: *up to 20 hours per week*.

**Studentship Regulations**: minimum number of teaching hours the student has to deliver: *at least 30 hours per week*.

\(^1\)Detailed changes can be found on the UK Home Office website under http://www.ukba.homeoffice.gov.uk/sitecontent/documents/news/sop4.pdf.
Modelling of Legal Specification

1 Motivation

2 Modelling of Legal Specification
   - Formal Model Sketch
   - Mapping to AnsProlog

3 Automatic Conflict Detection

4 Conflict Resolution via Automatic Revision

5 Future Work
Model sketch

Legal Specification

\[ \text{fluent}_1 \xrightarrow{\text{act}_1} \text{fluent}'_1 \xrightarrow{\text{act}_2} \text{fluent}''_1 \]

World Model

\[ \text{ObsEv}_1 \xrightarrow{} \text{ObsEv}_2 \xrightarrow{} \text{ObsEv}_3 \xrightarrow{} \text{ObsEv}_4 \]

- Model generates ordered traces that show us the evolution of the legal specification over time—allows validation and verification.
- Essential elements of model are:

Legal Specification: \( \mathcal{L} := \langle \mathcal{E}, \mathcal{F}, \mathcal{G}, \mathcal{C}, \Delta \rangle \)
Model sketch

Model generates ordered traces that show us the evolution of the legal specification over time—allows validation and verification.

Essential elements of model are:
- events ($\mathcal{E}$): exogenous and legal
- fluents ($\mathcal{F}$): power, permission, obligation, domain

Legal Specification: $\mathcal{L} := \langle \mathcal{E}, \mathcal{F}, \mathcal{G}, \mathcal{C}, \Delta \rangle$
Model sketch

Model generates **ordered traces** that show us the evolution of the legal specification over time—allows validation and verification.

Essential elements of model are:
- events ($\mathcal{E}$): exogenous and legal
- fluents ($\mathcal{F}$): power, permission, obligation, domain

Legal Specification: $\mathcal{L} := \langle \mathcal{E}, \mathcal{F}, G, \mathcal{C}, \Delta \rangle$
Model sketch

Model generates **ordered traces** that show us the evolution of the legal specification over time—allows validation and verification.

Essential elements of model are:
- events ($\mathcal{E}$): exogenous and legal
- fluents ($\mathcal{F}$): power, permission, obligation, domain

**Legal Specification:** $\mathcal{L} := \langle \mathcal{E}, \mathcal{F}, \mathcal{G}, \mathcal{C}, \Delta \rangle$
Model sketch

Model generates **ordered traces** that show us the evolution of the legal specification over time—allows validation and verification.

Essential elements of model are:
- events \( (E) \): exogenous and legal
- fluents \( (F) \): power, permission, obligation, domain

**Legal Specification**: \( \mathcal{L} := \langle E, F, G, C, \Delta \rangle \)
Framework translation to *AnsProlog*

- Answer set programming used as computational back-end
- **Conclusion**: Conditions with use of negation as failure
- Important components:
  - `observed(Event,Instant)`: an exogenous event at time *t*
  - `occurred(Event,Instant)`: a legal action at time *t*
  - `holdsat(Fluent,Instant)`: fluent is true at time *t*
  - `pow(Event),perm(Event),obl(Event,DueEvent,VioEvent)`: fluents that indicate norms
  - `initiated(Fluent,Instant)`: fluents to be added to state
  - `terminated(Fluent,Instant)`: fluents to be deleted from the state
Framework translation to *AnsProlog*

- Answer set programming used as computational back-end
- *Conclusion*: \(-\text{Conditions}\). with use of negation as failure
- Important components:
  - \text{observed}(\text{Event}, \text{Instant})\): an exogenous event at time \(t\)
  - \text{occurred}(\text{Event}, \text{Instant})\): a legal action at time \(t\)
  - \text{holdsat}(\text{Fluent}, \text{Instant})\): fluent is true at time \(t\)
  - \text{pow}(\text{Event}),\text{perm}(\text{Event}),\text{obl}(\text{Event}, \text{DueEvent}, \text{VioEvent})\): fluents that indicate norms
  - \text{initiated}(\text{Fluent}, \text{Instant})\): fluents to be added to state
  - \text{terminated}(\text{Fluent}, \text{Instant})\): fluents to be deleted from the state
Framework translation to *AnsProlog*

- Answer set programming used as computational back-end
- *Conclusion* : – *Conditions*. with use of negation as failure
- Important components:
  - observed(Event,Instant): an exogenous event at time $t$
  - occurred(Event,Instant): a legal action at time $t$
  - holdsat(Fluent,Instant): fluent is true at time $t$
  - pow(Event),perm(Event),obl(Event,DueEvent,VioEvent): fluents that indicate norms
  - initiated(Fluent,Instant): fluents to be added to state
  - terminated(Fluent,Instant): fluents to be deleted from the state
1 Motivation

2 Modelling of Legal Specification

3 Automatic Conflict Detection

4 Conflict Resolution via Automatic Revision

5 Future Work
Composite Legal Specification

- Composite Legal Specification $C_L$ with $\mathcal{L} = \{L_1, \ldots, L_n\}$
- Wrapper for independent legal specification. Individual state transition.
- Composite trace = sequence of exogenous events from all individual legal specification.
- Synchronised Traces: Null events ($\text{enull}$) to fill up the unrecognised events in individual traces. DOES NOT change the states.
Composite Legal Specification

- Composite Legal Specification $L = \{L_1, \ldots, L_n\}$
- Wrapper for independent legal specification. Individual state transition.
- Composite trace = sequence of exogenous events from all individual legal specification.
- Synchronised Traces: Null events ($e_{null}$) to fill up the unrecognised events in individual traces. DOES NOT change the states.
Composite Legal Specification

- Composite Legal Specification $C_L$ with $\mathcal{L} = \{\mathcal{L}_1, \ldots, \mathcal{L}_n\}$
- Wrapper for independent legal specification. Individual state transition.
- Composite trace = sequence of exogenous events from all individual legal specification.
- Synchronised Traces: Null events ($\text{enull}$) to fill up the unrecognised events in individual traces. DOES NOT change the states.
Conflicts occur when any two legal specifications disagree on a fluent known to both of them at the same time instant:

\[
\text{conflict}(FX, FY, I) : -\text{holdsat}(FX, I), \textbf{not} \ \text{holdsat}(FY, I), \\
\text{rename}(F, FX, X), \text{rename}(F, FY, Y), \\
\text{ifluent}(FX), \text{ifluent}(FY), \\
\text{instant}(I), \text{inst}(X; Y).
\]

Renaming: Contradiction results in no answer set.
Conflict traces

- Conflicts occur when any two legal specifications disagree on a fluent known to both of them at the same time instant:

  \[
  \text{conflict}(FX, FY, I) : \neg \text{holdsat}(FX, I), \textbf{not} \ \text{holdsat}(FY, I), \\
  \text{rename}(F, FX, X), \text{rename}(F, FY, Y), \\
  \text{ifluent}(FX), \text{ifluent}(FY), \\
  \text{instant}(I), \text{inst}(X; Y).
  \]

- Renaming: Contradiction results in no answer set.
Conflicts Detection

\[ CTR = \langle \ldots, sendOfferLetter, acceptOffer, askStudentship, applyVisa, \ldots \rangle \]

University Regulations:

UK Immigration Law:

\[
\begin{align*}
\text{conflict}(perm(\text{workUNI}(\text{tingting}, 30)), \text{permVI}(\text{tingting}, 30), 4) \\
\text{conflict}(perm(\text{workVI}(\text{tingting}, 20)), \text{permUNI}(\text{tingting}, 20), 4)
\end{align*}
\]
Automated Conflict Detection

Conflicts Detection

\[ CTR = \langle \ldots, sendOfferLetter, acceptOffer, askStudentship, applyVisa, \ldots \rangle \]

University Regulations:

- \( S_0 \) sendOfferLetter(tingting)
- \( S_1 \) acceptOffer(tingting)
- \( S_2 \) askStudentShip(tingting)
- \( S_3 \) enull

\( S_0 \) personUNI(tingting,overseas)
\( \text{perm}(sendOfferLetterUNI(tingting)) \)
\( \text{live}(\text{university}) \)
\( \text{perm}(acceptOfferUNI(tingting)) \)

\( S_1 \) personUNI(tingting,overseas)
\( \text{perm}(sendOfferLetterUNI(tingting)) \)
\( \text{live}(\text{university}) \)
\( \text{perm}(askStudentShipUNI(tingting)) \)
\( \text{perm}(acceptOfferUNI(tingting)) \)

\( S_2 \) personUNI(tingting,overseas)
\( \text{perm}(sendOfferLetterUNI(tingting)) \)
\( \text{live}(\text{university}) \)
\( \text{obl}(\text{workUNI}(tingting,30), \text{weekEnd}, \text{withdrawStudentshipUNI}) \)
\( \text{perm}(\text{workUNI}(tingting,30)) \)

\( S_3 \) personUNI(tingting,overseas)
\( \text{perm}(sendOfferLetterUNI(tingting)) \)
\( \text{live}(\text{university}) \)
\( \text{obl}(\text{workUNI}(tingting,30), \text{weekEnd}, \text{withdrawStudentshipUNI}) \)
\( \text{perm}(\text{workUNI}(tingting,30)) \)

UK Immigration Law:

- \( S_0 \) enull
- \( S_1 \) acceptOffer(tingting)
- \( S_2 \) enull
- \( S_3 \) applyVisa(tingting)

\( S_0 \) personVI(tingting,overseas)
\( \text{perm}(\text{acceptOfferVI}(tingting)) \)
\( \text{live}(\text{visa}) \)

\( S_1 \) live(\text{visa})
\( \text{perm}(\text{applyVisaVI}(tingting)) \)
\( \text{perm}(\text{acceptOfferVI}(tingting)) \)
\( \text{personVI}(tingting,overseas) \)
\( \text{obl}(\text{applyVisaVI}(tingting), \text{arrivalVI}(tingting), \text{illegalImmigrantVI}(tingting)) \)

\( S_3 \) live(\text{visa})
\( \text{perm}(\text{applyVisaVI}(tingting)) \)
\( \text{perm}(\text{acceptOfferVI}(tingting)) \)
\( \text{personVI}(tingting,overseas) \)
\( \text{obl}(\text{applyVisaVI}(tingting), \text{arrivalVI}(tingting), \text{illegalImmigrantVI}(tingting)) \)

\( S_4 \) live(\text{visa})
\( \text{perm}(\text{applyVisaVI}(tingting)) \)
\( \text{perm}(\text{acceptOfferVI}(tingting)) \)
\( \text{personVI}(tingting,overseas) \)
\( \text{studentVisaVI}(tingting,\text{tier4}) \)

conflict(\text{perm}(\text{workUNI}(tingting,30)), \text{permVI}(tingting,30), 4)
conflict(\text{perm}(\text{workVI}(tingting,20)), \text{permUNI}(tingting,20), 4)
Conflicts Detection

\[ CTR = \langle \ldots, \text{sendOfferLetter}, \text{acceptOffer}, \text{askStudentship}, \text{applyVisa}, \ldots \rangle \]

**University Regulations:**

- \(S_0\) \(\rightarrow\) \(S_1\)
  - person\(\text{UNI}(\text{tingting}, \text{overseas})\)
  - perm\(\text{sendOfferLetterUNI}(\text{tingting})\)
  - live(\text{university})
  - perm\(\text{acceptOfferUNI}(\text{tingting})\)

- \(S_1\) \(\rightarrow\) \(S_2\)
  - perm\(\text{sendOfferLetterUNI}(\text{tingting})\)
  - person\(\text{UNI}(\text{tingting}, \text{overseas})\)
  - perm\(\text{askStudentshipUNI}(\text{tingting})\)

- \(S_2\) \(\rightarrow\) \(S_3\)
  - perm\(\text{acceptOfferUNI}(\text{tingting})\)
  - person\(\text{UNI}(\text{tingting}, \text{overseas})\)
  - perm\(\text{workUNI}(\text{tingting}, 30)\)

- \(S_3\) \(\rightarrow\) \(S_4\)
  - perm\(\text{workUNI}(\text{tingting}, 30)\)

**UK Immigration Law:**

- \(S_0\) \(\rightarrow\) \(S_1\)
  - person\(\text{VI}(\text{tingting}, \text{overseas})\)
  - perm\(\text{acceptOfferVI}(\text{tingting})\)

- \(S_1\) \(\rightarrow\) \(S_2\)
  - live(\text{visa})
  - perm\(\text{applyVisaVI}(\text{tingting})\)
  - perm\(\text{acceptOfferVI}(\text{tingting})\)

- \(S_2\) \(\rightarrow\) \(S_3\)
  - live(\text{visa})
  - perm\(\text{applyVisaVI}(\text{tingting})\)
  - perm\(\text{workVI}(\text{tingting}, 20)\)

- \(S_3\) \(\rightarrow\) \(S_4\)
  - live(\text{visa})
  - perm\(\text{applyVisaVI}(\text{tingting})\)
  - student\(\text{VisaVI}(\text{tingting}, \text{tier4})\)

\[ \text{conflict}(\text{perm(\text{workUNI}(\text{tingting}, 30)), perm\text{VI}(\text{tingting}, 30), 4}) \]
\[ \text{conflict}(\text{perm(\text{workVI}(\text{tingting}, 20)), perm\text{UNI}(\text{tingting}, 20), 4}) \]
Motivation

Modelling of Legal Specification

Automatic Conflict Detection

Conflict Resolution via Automatic Revision

Future Work
Theory Revision Task is characterized by a tuple $\langle P, B, T, M \rangle$ and $T'$, called revised theory, is a solution to the task with cost $c(T, T')$, iff:

1. $T' \subseteq s(M)$,
2. $B \cup T' \models P$,
3. $c(T, T')$ is minimal.

Given a trace $CTR$ which admits a conflict $c = \text{conflict}(FX, FY, I)$ and assume $\mathcal{L}_X \succ \mathcal{L}_Y$:

Inductive Logic Programming produces all possible revisions $\mathcal{L}'_Y$:

$$\mathcal{L}'_Y \cup (\mathcal{C}_\mathcal{L} \setminus \mathcal{L}_Y) \cup CTR \models \neg c$$

---

Conflict Resolution via Theory Revision

- Theory Revision Task is characterized by a tuple $\langle P, B, T, M \rangle$ and $T'$, called revised theory, is a solution to the task with cost $c(T, T')$, iff
  1. $T' \subseteq s(M)$,
  2. $B \cup T' \models P$,
  3. $c(T, T')$ is minimal.

- Given a trace $CTR$ which admits a conflict $c = \text{conflict}(FX, FY, I)$ and assume $L_X \succ L_Y$:

  $\begin{align*}
  \langle P, & B, T, M \rangle \\
  \downarrow & \downarrow \downarrow \downarrow \\
  \neg c & C_L \setminus L_Y L_Y M_{C_L}
  \end{align*}$

- Inductive Logic Programming produces all possible revisions $L_Y'$:

  $L_Y' \cup (C_L \setminus L_Y) \cup CTR \models \neg c$

---

Theory Revision Task is characterized by a tuple $\langle P, B, T, M \rangle$ and $T'$, called revised theory, is a solution to the task with cost $c(T, T')$, iff

1. $T' \subseteq s(M)$,
2. $B \cup T' \models P$,
3. $c(T, T')$ is minimal.

Given a trace $CTR$ which admits a conflict $c = conflict(FX, FY, I)$ and assume $\mathcal{L}_X \succ \mathcal{L}_Y$:

\[
< P, \hspace{1cm} B, \hspace{1cm} T, \hspace{1cm} M > \downarrow \downarrow \downarrow \downarrow \\
\neg c \hspace{0.5cm} C_L \setminus \mathcal{L}_Y \hspace{0.5cm} \mathcal{L}_Y \hspace{0.5cm} M_{C_L}
\]

Inductive Logic Programming produces all possible revisions $\mathcal{L}'_Y$:

\[
\mathcal{L}'_Y \cup (C_L \setminus \mathcal{L}_Y) \cup CTR \models \neg c
\]

---

Case Study

Rule 2: \( \text{initiated}(\text{perm}(\text{work(Student, 30)}, I)): - \\
\text{occurred}(\text{askStudentShip(Student), I}), \\
\text{holdsat}(\text{availability, I}), \\
\text{not holdsat}(\text{person(Student, overseas), I}). \)

Rule 3: \( \text{initiated}(\text{obl}(\text{work(Student, 30), weekEnd, withdrawStudentship), I)): - \\
\text{occurred}(\text{askStudentShip(Student), I}), \\
\text{holdsat}(\text{availability, I}), \\
\text{not holdsat}(\text{person(Student, overseas), I}). \)
Case Study

**Rule 2**: initiated\(perm(work(Student, 30)), I)\): —
occurred\(askStudentShip(Student), I)\),
holdsat\(availability, I)\),
\textbf{not} holdsat\(person(Student, overseas), I)\).

**Rule 3**: initiated\(obl(work(Student, 30), weekEnd, withdrawStudentship), I)\): —
occurred\(askStudentShip(Student), I)\),
holdsat\(availability, I)\),
\textbf{not} holdsat\(person(Student, overseas), I)\).
Future Work

1 Motivation

2 Modelling of Legal Specification

3 Automatic Conflict Detection

4 Conflict Resolution via Automatic Revision

5 Future Work
Future Work

- Measurement Mechanism of Revision Cost
- Ontology Alignment
- Interfaces & Tools
Thank you for your attention!
Any questions?